Data Distribution & Processing CSCI
Constraint Management CSC
THOR 3 Design Panel 3

84K00530-070

May 14, 1998
Version 1.1

1. Data Distribution & Processing CSCI

The Data Distribution & Processing CSCI is composed of the following CSCs:
Data Distribution CSC, Data Fusion CSC, Data Health CSC, and Constraint Management CSC.

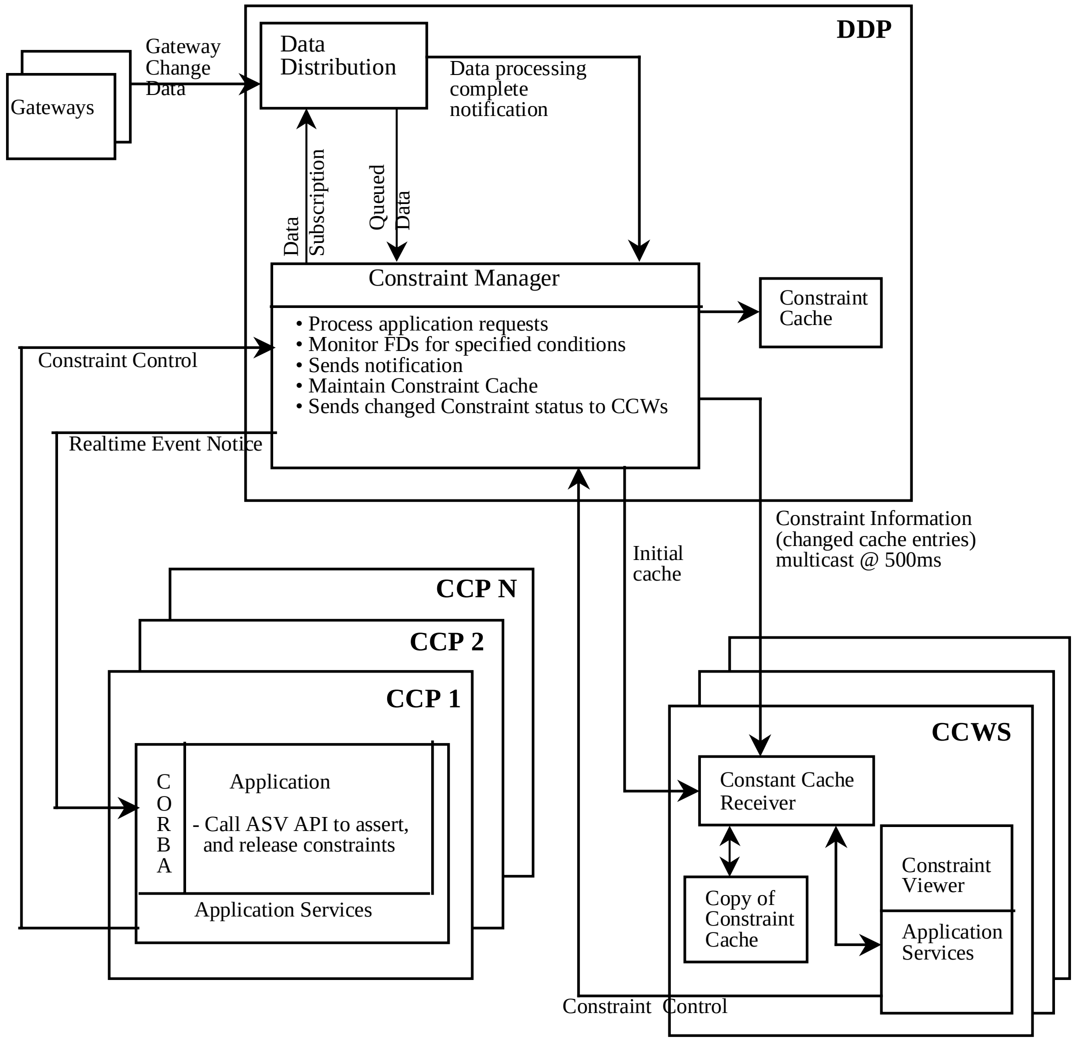
1.1 Constraint Management (CMS) CSC Introduction

1.1.1 Constraint Management CSC Overview

The Constraint Management (CMS) CSC provides the capability to monitor one or more FDs for a predetermined condition and notify personnel operating the Test Set, and software applications executing within the Test Set, that the monitored data has transition into or out of the constraint condition.

Constraint Manager CSC Overview is as follows:

\



1.1.2 Constraint Management CSC Operational Description

The Constraint Management CSC supports asserting constraints against FDs defined in the OLDB. The applications/viewers are notified of the exceptions via constraint notification events.

1.2 Constraint Management CSC Specifications

1.2.1 Constraint Management CSC Ground Rules

1. Redundancy management will not be supported (future requirement).
2. Persistent data/checkpoint will not be supported (future requirement).
3. The constraint criteria can only be specified during run-time (predefined criteria is a future requirement).
4. All logging to the SDC will be done by System Services.
5. Notification of failed applications will be performed by System Integrity.

1.2.2 Constraint Management CSC Functional Requirements

1.2.2.1 Constraint Management (CMS) at the DDP

1. CMS will provide the capability to maintain the constraint cache.
2. CMS will provide the capability to update the constraint cache with the following information:
   a. Constraint ID
      Unique identifier assigned by the Constraint Manager
   b. FDID
      Identifier of the FD associated with the Constraint
   c. Owner
      - CPU ID
        CCWS/CCP that invoked the assertion
      - Application ID
        Application/Viewer identifier that invoked the assertion
      - Process ID
        Application/Viewer unique process id that invoked the assertion
   d. Value State Check
      - Algorithm Identifier
        If set, identifier for an algorithm to be performed based on the type of the FD (analog, discrete, digital pattern, or enumeration)
      - Input Arguments
        Defined based on algorithm (eg limits, mask, normal state)
      - State Notification Flag
        Application wants notification if the value goes into exception
   e. Boundary Check Flag
      If specified, monitor to see if the exception is maintained over a period of time or for a number of samples (this is optional)

f. Viewability Flag

Flag indicating cache notification events can be displayed by Viewers

g. FD Value

Value of the FD at the time of the exception

h. Category

Category type of constraint assertion

example:

RCL	- Reactive Control Logic
LCC	- Launch Commit Criteria
OMRSD	-
Control	- Used by EIM for control

i. User Class

Used to authenticate commands

j. Inhibit Flag

Indicates the constraint algorithms will not be performed for changed data

k. Notification Sent

False - indicates either no notification was sent

True - indicates a notification event was sent to the asserting application, which includes state change, alter, inhibit, and release.

m. Update Count

Number of times the constraint entry has been modified

n. Current Time

Time stamp associated with the FD for the constraint

o. Current State and Sub-state

State of current constraint (Violated, Normal, Invalid)

State	**Sub-State**
Invalid	Inhibited Failed Health
Violated	Low High Delta Equal NotEqual Period Sample In List Out of List Voting Change Invalid Override
Normal	Data Change in Limits Invalid Override

p. Previous Time

Time stamp of previous constraint state change

q. Previous State and Sub-state

Previous state of constraint (see Current State)

r. Exception Count

Number of transitions to exception that have occurred

3. CMS will make use of Data Distribution API to obtain constraint information.
4. CMS will make information available for application access at the CCWS via Application Services.
5. CMS will provide the capability to receive altered constraint entries from the CCP and CCWS.
6. CMS will provide the capability to distribute the constraint updates to DCN at a 500 ms rate using Reliable Multicast.
7. CMS will provide the capability to distribute the constraint notification event messages to the DCN at a 500 ms rate using reliable multicast.
8. CMS will provide the capability to check FD values against the constraint criteria.
9. CMS will provide the capability to notify the asserting application with an event change when one of the following conditions occur:

- one of the following asserted conditions change state:

  for analog measurements:

  a) test upper limit
  b) test lower limit
  c) delta change

  for discrete measurements:

  a) any change in value

  for digital pattern measurements:

  a) test lower limit
  b) test upper limit
  c) equal to
  d) not equal to
  e) delta change

  for enumerated measurements:

  a) equal to
  b) not equal to
  c) in list
  d) not in list

- a health change of state

  For any of the asserted conditions, a check will be performed which will generate a notification if the health fails or returns to normal

- voting change

  When one or more of the members of a compound constraint caused a state change

- a constraint exceeds the boundary check

  When a constraint is in exception over a period of time or for a number of samples

10. CMS will provide the capability to include the following information in the event notification:

• Constraint ID	- Unique constraint identifier assigned by Constraint Manager
• Time Stamp	- System time that the constraint change occurred
• Event_ID constraint	- Unique Event ID used to identify relationships to a summary
• State	- Current state at the time of constraint state change (invalid, normal, violated)
• Sub-State	- Current sub-state at the time of constraint state change (high, low, normal, inhibit-override, etc)
• Value	- FD value at the time the constraint change occurred
• Health	- Current health at the time the constraint change occurred
• Count	- number of violations
• FDname	- FD name
• Category	- Application Attributes

11. CMS will provide the capability to notify the asserting application with a state change when one of the following conditions occur:

- a constraint is altered

  When an application alters a constraint, the asserting application is notified on the current contents of the constraint cache entry
- a constraint is released

  When an application releases a constraint, the asserting application is notified that the constraint has been released.

12. CMS will provide the capability to include the following information in the state notification:

• Constraint ID	- Unique constraint identifier assigned by Constraint Manager
• Event_ID constraint	- Unique Event ID used to identify relationships to a summary
• Condition	- Asserted, Altered, Released, Inhibited, or Activated
• FDname	- FD name

13. CMS will provide the capability to authenticate constraint commands
14. CMS will release any constraints for applications abnormally terminating

1.2.2.2 Constraint Management at the CCP

1. CMS will provide the capability to output altered constraint specifications to the DDP.
2. CORBA will provide the capability to receive real time events from the DDP and deliver them to the proper applications.

1.2.2.3 Constraint Management at the CCWS

1. CMS will provide the capability to output altered constraint entries to the DDP.
2. CMS will provide the capability to receive altered constraint updates on the DCN from the DDP.
3. CMS will provide the capability to maintain a copy of the constraint cache.
4. CMS will provide the capability to filter the constraint changes on one or more of the following fields:

• FDID	- Identifier of the FD
• Category	- Category for constraint (RCL, LCC, etc)

• Owner	- Application/Viewer which invoked constraint
• User Class	- User Class for constraint
• Time Span	- Range of time for constraints

NOTE: Only constraint messages where the viewability flag is set and where the user class of the constraint message is enabled at the CCWS will be made available to the constraint message viewer, regardless of other filter settings. Filters may be set to additionally restrict which entries will be displayed.

1.2.2.4 Constraint Management Application Interfaces

1. Application interfaces for performing constraint commands are as follows:

   a) ASSERT_ATOMIC **-** Provides the capability to specify the conditions upon which an event notification will be triggered when the conditions are met.

Request (CCP/CCWS):	Application Response (Asserting Application):
FD Name	Status
Algorithm	Error Code
Health Check	Constraint ID
Boundary Check	
Category	
Viewability	
User Class	
Input Arguments	

   b) ASSERT_COMPOUND – Provides the capability to specify a compound constraint to perform voting based on algorithmic combinations of multiple constraints as members of a compound constraint expression.

Request (CCP/CCWS):	Application Response (Asserting Application):
Expression	Status
	Error Code
	Constraint ID

   c) ASSERT_SUMMARY – Provides the capability to specify a summary ompound constraint to perform voting based on algorithmic combinations of multiple constraints as members of a compound constraint expression.

Request (CCP/CCWS):	Application Response (Asserting Application):
Members	Status
	Error Code
	Constraint ID

   d) ASSERT_FILE **-** Provides the capability to specify a file containing the assertion requests

   Notification will be triggered when the conditions are met for any of the constraints in the file

Request (CCP/CCWS):	Application Response (Asserting Application):
File Name	Status
	Error Code

   e) RELEASE **-** Provides the capability for a user and application to dynamically release a constraint.

Request (CCWS):	Response (Interrogating Application):
Constraint ID	Status
User Class	Error Code

   f) RELEASE_FILE **-** Provides the capability to specify a file of constraints to be released.

Request (CCWS):	Response (Interrogating Application):
File Name	Status
User Class	Error Code

   g) ALTER **-** Provides the capability for a user and application to dynamically modify the Viewability Flag, Input Arguments, and the Notification Flags on an active constraint.

Request (CCP/CCWS):	Response (Asserting Application):
Constraint ID	Status

User Class	Error Code
Alter Option	
Alter Field Values	

h) INHIBIT - Provides the capability for a user to inhibit the processing of the constraint algorithm.

<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
User Class	Error Code

i) INHIBIT_WITH_OVERRIDE - Provides the capability for a user to inhibit the processing of the constraint algorithm and override the current state

<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
User Class	Error Code
NewState	

j) ACTIVATE - Provides the capability for a user to activate the processing of the constraint algorithm.

<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
User Class	Error Code

k) REGISTER_CALLBACKS - Provides the capability for a user to register to receive event and state notifications.

<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
EventCB	Error Code
StateCB	

l) DEREGISTER_CALLBACKS - Provides the capability for a user to de-register from receiving realtime event notifications.

<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
User Class	Error Code

m) RETRIEVE_RCL - Provides the capability for a user to retrieve the RCL constraint associated with an FDID.

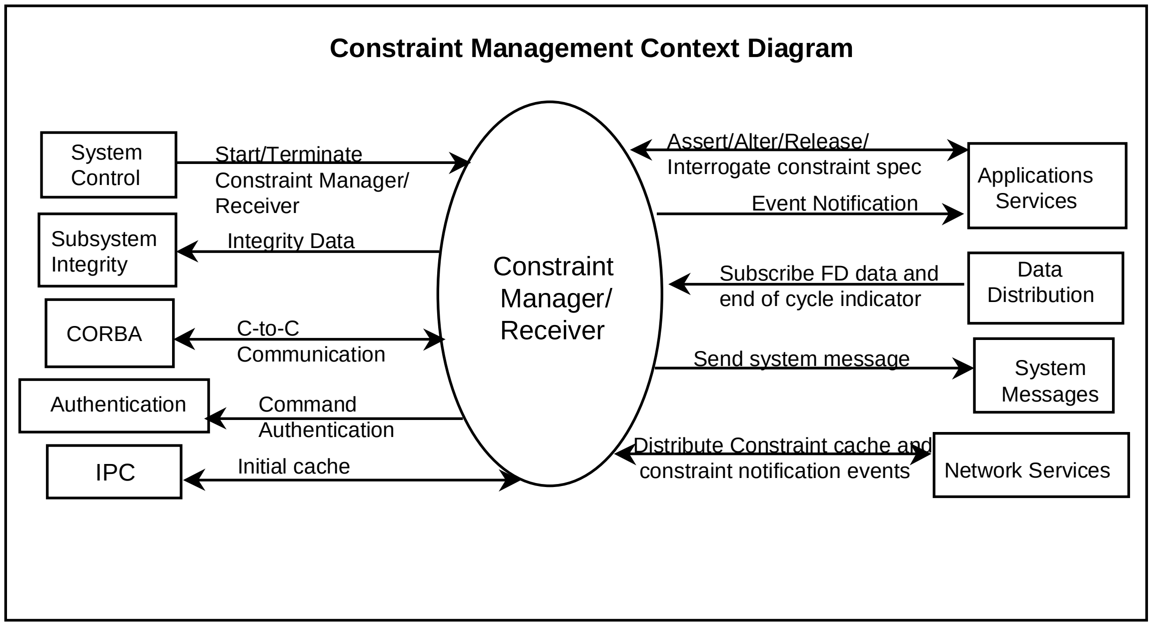
<u>Request (CCP/CCWS):</u>	<u>Response (Asserting Application):</u>
Constraint ID	Status
FDID	Error Code

2. All constraint commands, responses, and notifications will be logged to the SDC.
3. Application interfaces for retrieving the constraint cache will be based on the applied filters.

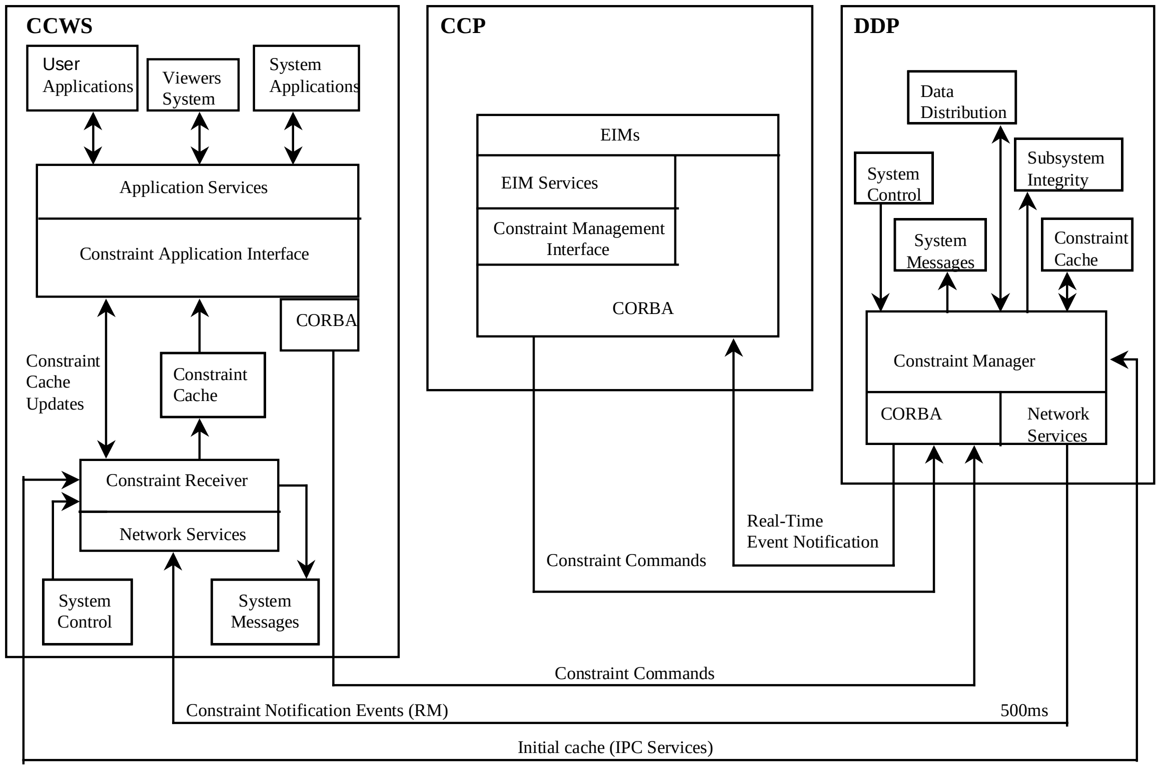
1.2.3 Constraint Management CSC Performance Requirement

CMS will provide the capability to process all constraints at the System Synchronous Rate (SSR).

1.2.4 Constraint Management CSC Interfaces


Constraint Management Context Diagram
System Control
Start/Terminate Constraint Manager/ Receiver
Subsystem Integrity
Integrity Data
CORBA
C-to-C Communication
Authentication
Command Authentication
IPC
Initial cache
Constraint Manager/ Receiver
Assert/Alter/Release/ Interrogate constraint spec
Applications Services
Event Notification
Subscribe FD data and end of cycle indicator
Data Distribution
Send system message
System Messages
Distribute Constraint cache and constraint notification events
Network Services

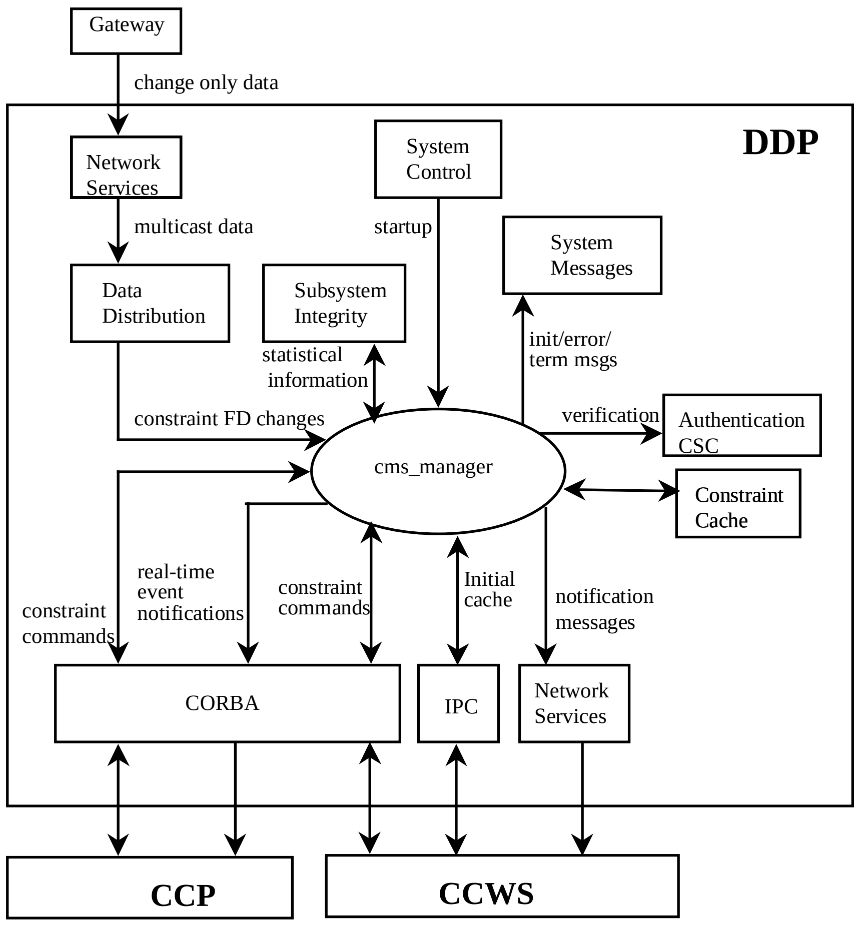
1.2.5 Constraint Management CSC Data Flow Diagram



1.3 Constraint Management CSC Specifications

1.3.1 Constraint Management Detailed Data Flow

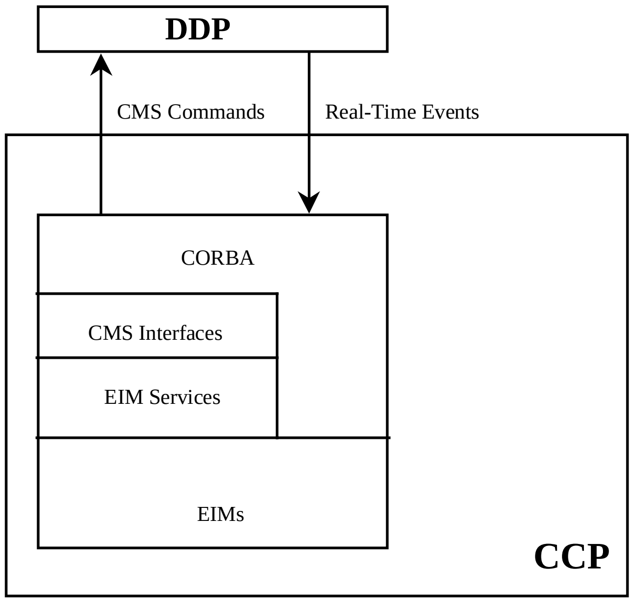
1.3.1.1 DDP Detailed Data Flow



The Constraint Management CSC on the DDP will perform the following functions:

1. Perform constraint algorithms when the data values change for the associated FD
2. Issue real-time notification to asserting application on the CCP.
3. Issue constraint cache notification events to the CCWSs.
4. Output notifications to the CCWS at 500ms.
5. Maintain the constraint cache table.

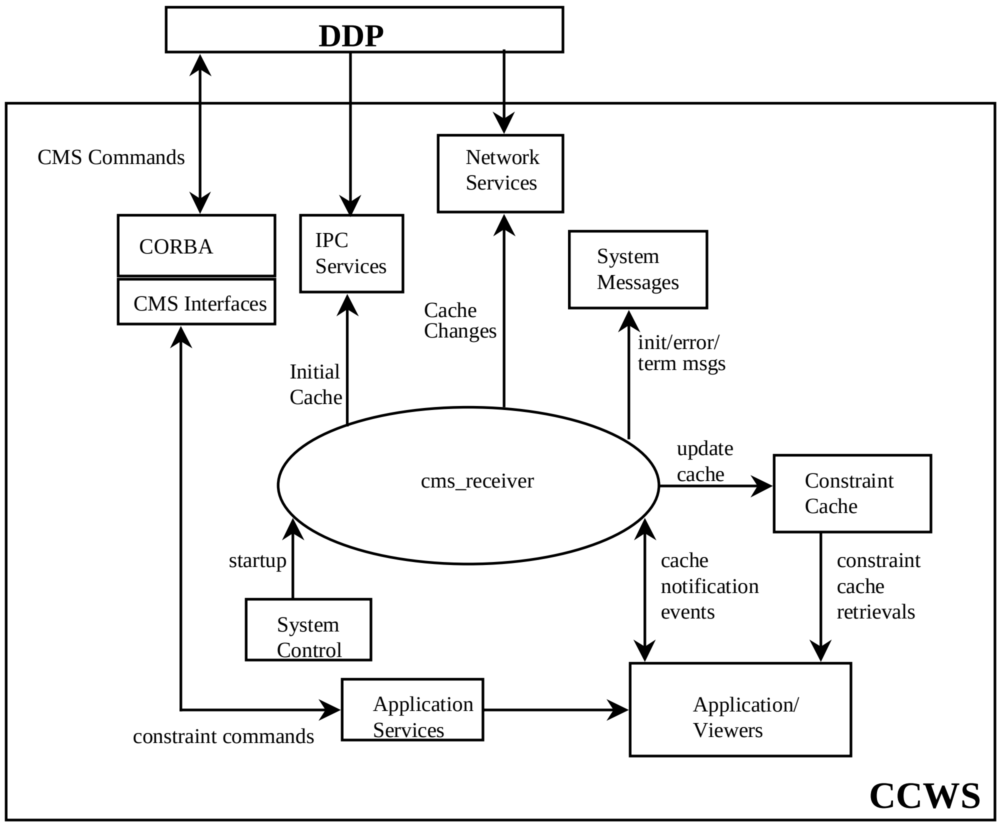
1.3.1.2 CCP Detailed Data Flow



The Constraint Management CSC on the CCP will perform the following functions:

1. Perform assertions on constraints.
2. Receive real-time notifications.

1.3.1.3 CCWS Detailed Data Flow



The Constraint Management CSC on the CCWS will perform the following functions:

1. Perform assertions on constraints.
2. Receive all constraint change events via network services.
3. Provide the capability to interrogate constraint information.

1.3.2 Constraint Management External Interfaces

1.3.2.1 Constraint Management Message Formats

1.3.2.1.1 cms_manager messages

1. Message Group = CMS
Severity = Informational
CMS manager is initialized
Help Information Content:
The manager process has initialized successfully.
Detailed Information:
N/A

2. Message Group = CMS
Severity = Error
CMS manager was unable to open the #ARGUMENT1# multicast address, errno = #ARGUMENT2#
Help Information Content:
The manager process was unable to open the address for the constraint cache updates.
Detailed Information:
N/A

3. Message Group = CMS
Severity = Error
**CMS manager was unable to open the CORBA interface,
errno = #ARGUMENT1#**
Help Information Content:
The manager process was unable to open the CORBA interface.
Detailed Information:
N/A

4. Message Group = CMS
Severity = Error
CMS manager was unable to send a message via CORBA, errno = #ARGUMENT1#
Help Information Content:
The manager process was unable to send a request to CORBA.
Detailed Information:
N/A

5. Message Group = CMS
Severity = Error
CMS manager was unable to connect to data distribution, errno = #ARGUMENT1#
Help Information Content:
The manager process was unable to connect to data distribution to receive change data.
Detailed Information:
N/A

6. Message Group = CMS
Severity = Error
CMS manager was unable to register #ARGUMENT1# will CORBA
Help Information Content:

The manager process was unable to register the the CORBA interface.
Detailed Information:
N/A

7. Message Group = CMS
Severity = Informational
CMS manager has terminated
Help Information Content:
The receiver process has terminated successfully.
Detailed Information:
N/A

1.3.2.1.1 cms_rcvr messages

1. Message Group = CMS
Severity = Informational
CMS rcvr is initialized
Help Information Content:
The receiver process has initialized successfully.
Detailed Information:
N/A

2. Message Group = CMS
Severity = Error
CMS rcvr was unable to open the #ARGUMENT1# multicast address, errno = #ARGUMENT2#
Help Information Content:
The receiver process has initialized successfully.
Detailed Information:
N/A

3. Message Group = CMS
Severity = Informational
CMS rcvr has terminated
Help Information Content:
The receiver process has initialized successfully.
Detailed Information:
N/A

1.3.2.2 Constraint Management Display Formats

There are no display formats for the CMS CSC.

1.3.2.3 Constraint Management Input Formats

There are no input formats for the CMS CSC.

1.3.2.4 Constraint Management Recorded Data

Statistical data will be recorded in the Constraint Cache header. The following information will be stored:

- Number of active constraints
- Number of violations
- Number of assertions
- Number of notifications
- Number of updates (per constraint)
- Number of alters
- Number of releases
- Number of removes

1.3.2.4 Constraint Management Printer Formats

There are no printer formats for the CMS CSC.

1.3.2.4 Constraint Management Inter-process Communication

Inter-process communication is handled in IDD format messages as defined in the RTPS Packet Payload ICD, 84K00357-001 dated xxx 1997.

1.3.2.5 Constraint Manager External Interface Calls

1.3.2.5.1 Constraint Command Interfaces

a) ASSERT

Add a new constraint to be monitored. A constraint cache entry is generated and the current value for the FD is used to determine the initial constraint state.

Assert_Analog()

Assert_Discrete()

Assert_DigitalPattern()

Assert_Enumeration()

Assert_Compound()

Assert_Summary()

Assert_File()

b) RELEASE

Releases an asserted constraint from being monitored. The constraint cache entry is removed.

Release_By_RCL()

Release_By_CID()

c) ALTER

Alters the information for a currently asserted constraint

Alter_Analog()

Alter_Discrete()

Alter_DigitalPattern()

Alter_Enumeration()

Alter_Compound()

Alter_Summary()

d) INHIBIT

Inhibit an asserted constraint.

Inhibit_By_RCL()

Inhibit_By_CID()

e) ACTIVATE

Activate a constraint.

Activate_By_RCL()

Activate_By_CID()

f) REGISTER_CALLBACKS

Register to receive a real-time notification for an active constraint.

RegisterCB_By_RCL()

RegisterCB_By_CID()

g) DEREGISER_CALLBACKS

Deregister to receive a real-time notification for an active constraint.

DeregisterCB_By_RCL()

DeregisterCB_By_CID()

h) RETRIEVE_RCL

Retrieve the RCL constraint associated with an FDname.

Retrieve_RCL_Constraint()

1.3.2.5.2 Cache Notification Update Interfaces

a) apply_filter()

Applies filter criteria for interrogating and retrieving cache entries

b) Remove_filter()

Removes applied filters

c) interrogate()

Requests all cache entries matching the filter criteria.

d) Register_for_constraints()

Notifies constraint management to send filtered cache updates.

e) Deregister_for_constraints

Notifies Constraint Management to stop sending filtered cache updates

1.3.3 Constraint Manager Internal Interfaces

1.3.3.1 Constraint Cache Interfaces

Cache is maintained in local memory by the receive and manager processes. It contains a list of all the active constraints.

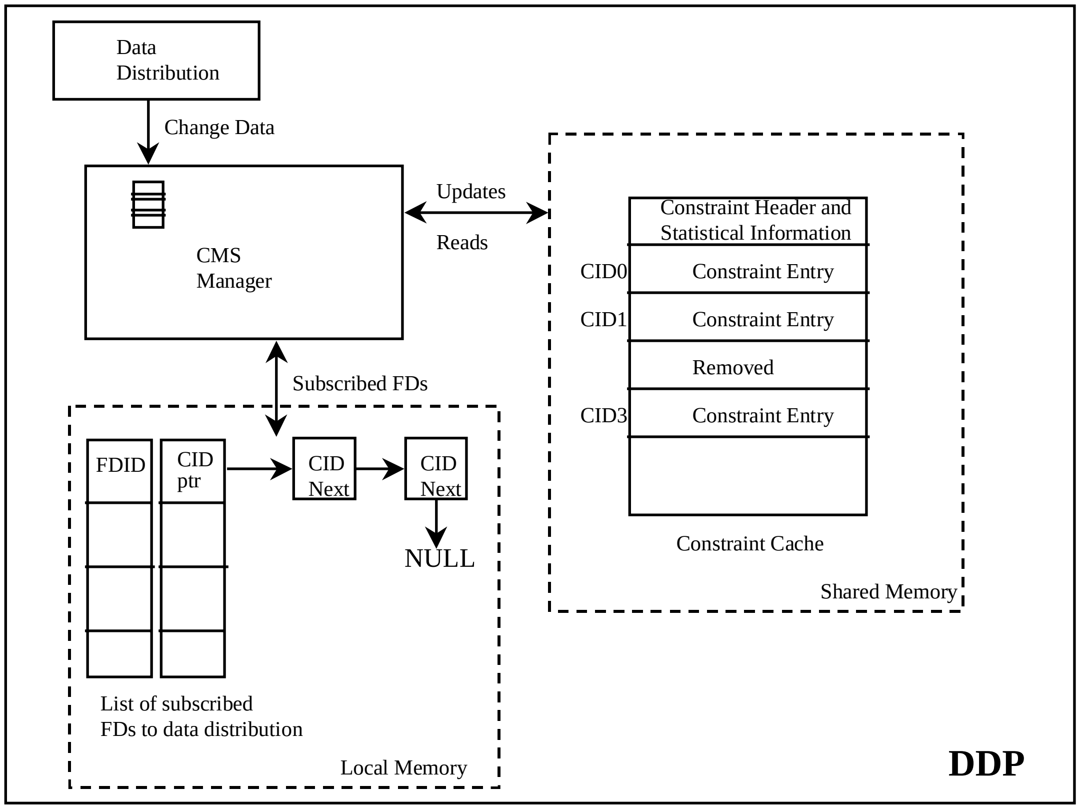
1.	cache_add()	- add a new entry into the constraint cache
2.	cache_remove()	- remove an existing entry in the constraint cache
3.	cache_read_first()	- read the first record in the cache
4.	cache_read_next()	- read the next record in the cache
5.	cache_write()	- write a modified entry into the constraint cache
6.	cache_initialize()	- initialize the constraint cache
7.	cache_set_filter()	- filter out fields for the read commands
8.	cache_remove_filter()	- remove the filters applied with the cache_set_filter command

1.3.3.2 Constraint Command Matrix

Command	**Method**	**EIM Notification**	**SDC Recorded**	**Performed at the DDP**
Assert	add	Async	Yes	Yes
Release	delete	Immediate	Yes	Yes
Interrogate	read	N/A	No	No
Wait_for_data	read next	N/A	No	No
Apply_filter	set_filter	N/A	No	No
Remove_filter	remove_filter	N/A	No	No
Alter	write	Immediate	Yes	Yes
Inhibit	inhibit	Immediate	Yes	Yes
Activate	inhibit	Immediate	Yes	Yes

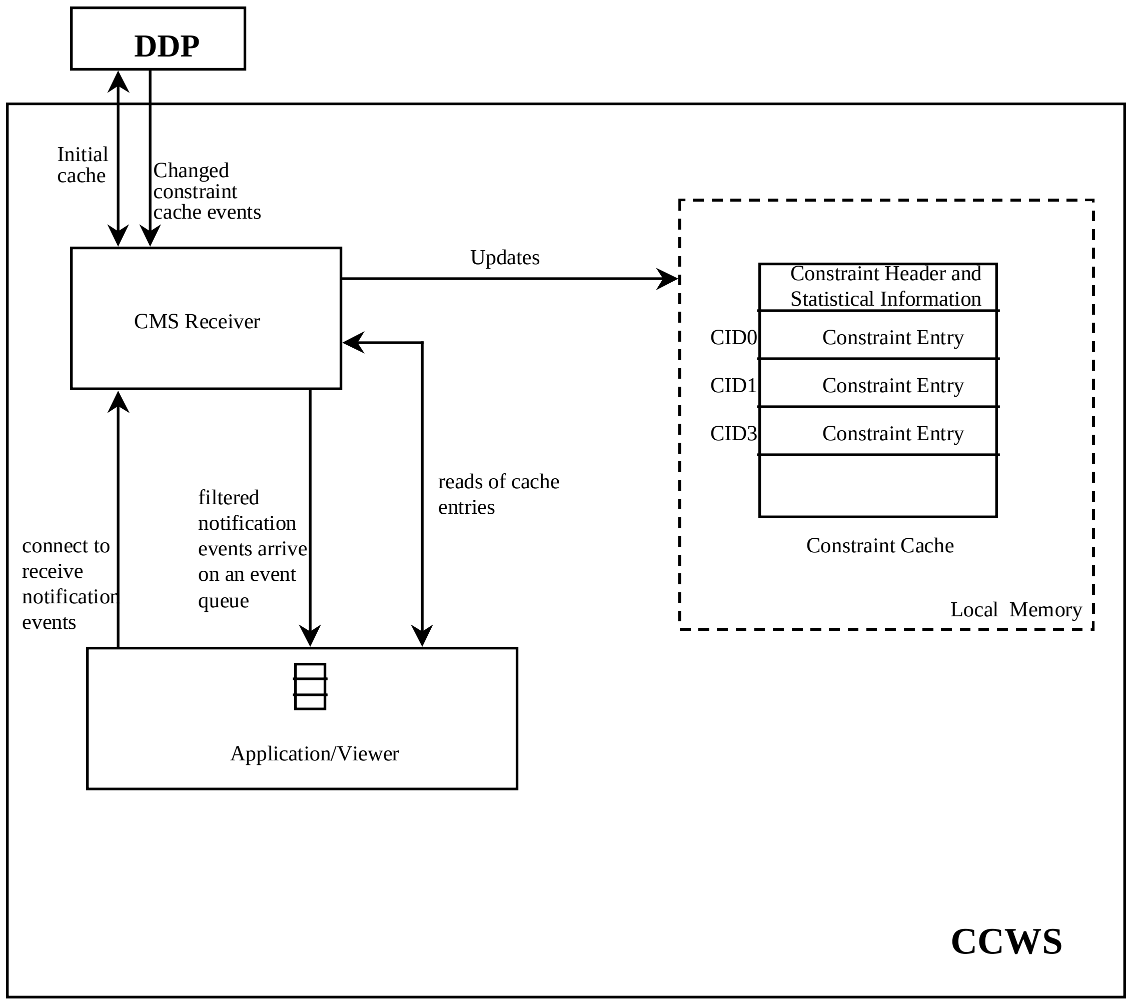
1.3.4 Constraint Manager Structure Diagram

1.3.4.1 Manager Object



The manager process maintains the global constraint cache list in shared memory. A local list is defined to contain the list of FDs associated with the constraints. This list is used for subscribing to Data Distribution for change only values. When change values arrive from Data Distribution, the constraint list is searched for the FD received. The associated constraints are used to perform the algorithms. If notification is to be sent to the asserting application, the application is notified with a real-time event (if it resides on the CCP), and a notification event is placed on the notification list to be sent to the CCWSs. The notification events are sent to the CCWSs even if the "viewability flag" is false, so the events will be logged to the SDC.

1.3.4.1 Receiver Object


DDP
Initial cache
Changed constraint cache events
CMS Receiver
Updates
Constraint Header and Statistical Information
CID0
Constraint Entry
CID1
Constraint Entry
CID3
Constraint Entry
Constraint Cache
Local Memory
reads of cache entries
filtered notification events arrive on an event queue
connect to receive notification events
Application/Viewer
CCWS

The cms_receiver process maintains a local copy of the constraint cache. Constraint cache notification events are received from the cms_manager process every 500ms. The events are read and filtered through the "filter criteria" specified by the connected applications. If the event passed the filter criteria as well as the viewability flag and user class, the event is placed on the application's queue. Once the events are processed, the applications which are to receive events, will have events placed on their socket and will be notified that there are new events to be read.

1.3.5 Constraint Manager Test Plan

1.3.5.1 Environment

Constraint Events will be monitored on the DDP and exceptions will be sent to both the CCP and the CCWS. EIM test tools will simulate End Item Managers, and the Constraint Management Viewer will be available for the CIT. The DDP, CCP, and CCWS will be needed for the test cases.

1.3.5.1 Test Tools

1. Test Data Generator - Generator for test data
2. FD_publisher - data distribution data publisher
3. EIM test tools - perform EIM assertions, alters, and releases.
4. CMS Viewer - performs Viewer assertions, alters, releases, and viewing of change events.

1.3.5.1 Test Cases

Test Case 1 - Determine when constraint conditions are exceeded
Test Approach Summary:
Provide the capability to monitor Measurement FDs at the rate the data changes and determine when predefined constraint limits are exceeded or constraint conditions are met. Use the Constraint Management test tools to assert, alter, and release constraints. Use the Data Distribution publishing tool to cause the FDs to go in and out of the different limits set. Use the Constraint Management test tools to view the constraint events.
Required Test Configuration / Dependencies:

- Ops CM to Configure DDP/CCWSs
- DDP and CCWS platforms
- RTCN and DCN
- Reliable Messaging / Network Services
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID
- Data Distribution / CVT on DDP and CCWS
- Data Distribution publish tool
- Constraint Management test tools

Test Case 2 - Test multiple users requesting notification of events
Test Approach Summary:
Provide the capability for multiple users and system or user applications to request notification of constraint events for each Measurement FD. Use multiple instances of the EIM test tool to assert constraints. Use the Data Distribution publishing tool to cause the FDs to go in and out of the different limits for the notifications to be set. Use the EIM test tool to view these notifications.
Required Test Configuration / Dependencies:

- Ops CM to Configure DDP/CCP
- DDP and CCP platforms
- RTCN
- Reliable Messaging / Network Services
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID
- Data Distribution / CVT on DDP and CCP
- EIM test tool
- Data Distribution publish tool

Test Case 3 - Test notification flags
Testing Approach Summary:
Provide the capability for each user, and system or user application requesting constraint notification to specify the limits/condition under which they will be notified. Use the EIM test tool to assert multiple constraints with various notification flags set. Use the Data Distribution publishing tool to cause the FDs to go in and out of the different limits for the notifications to be set. Use the EIM test tool to view these notifications.
Required Test Configuration / Dependencies:

- Ops CM to Configure DDP/CCP
- DDP and CCP platforms
- RTCN
- Reliable Messaging / Network Services
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID
- Data Distribution / CVT on DDP and CCP
- EIM test tool
- Data Distribution publish tool

Test Case 4 - Set more than one set of limits on an FD
Testing Approach Summary:
Allow an application to set more than one set of limits on a measurement FD. Use the Constraint Management test tools to assert multiple constraints against an FD. Use the Data Distribution publishing tool to cause the FDs to go in and out of the different limits set.
Required Test Configuration / Dependencies:

- Ops CM to Configure DDP/CCWSs
- DDP and CCWS platforms
- RTCN and DCN
- Reliable Messaging / Network Services
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID
- Data Distribution / CVT on DDP and CCWS
- Data Distribution publish tool
- Constraint Management test tools

Test Case 5 - Assert and View constraints
Summary of Testing Approach:
Use a viewer which provides a mechanism for asserting and viewing constraints against measurement FDs. Use the Constraint Management Viewer to assert constraints against multiple FDs. Use the Data Distribution publishing tool to cause the FDs to go in and out of exception and view the constraint events on the Viewer.
Required Test Configuration / Dependencies:

- Ops CM to Configure DDP/CCWSs
- DDP and CCWS platforms
- RTCN and DCN
- Reliable Messaging / Network Services
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID
- Data Distribution / CVT on DDP and CCWS
- Data Distribution publish tool
- Constraint Management Viewer

Appendix A

Statement Of Work (Thor)

- Define the list of *logical, mathematical*, and relational function required by the users for Constraint Management.
- Determine if a Control Shell can be utilized and provide the selected tool.
- *Provide the initial Pre-Build Constraint Management Editor.*
- Provide the capability for Fused FDs to be utilized by Constraint Management.
- Provide an initial API for System Viewers with the minimum capability to access Constraint.
- Provide performance data for system modeling.
- Confirm and/or modify system data flow diagrams for Constraint Management and provide the appropriate updates to the SDD.
- Define Packet formats for Constraint Management.
- Provide the capability to monitor Measurement FDs at the rate the data changes and determine when constraint conditions are met.
- Provide the capability for multiple users and system or user applications to request notification of constraint events for each Measurement FD.
- *The CLCS shall provide the capability to monitor measurement data (both converted count data or calibrated engineering units) for out of limits excursions and notify registered uses when any of the following conditions occurs:*
  - *N samples in a row that meet the constraint*
  - *N samples in a given time that meet the constraint*
  - *There is less than a specified time between constraint events*
- Provide the capability to access current constraints and their algorithms.
- Provide the capability to create new constraints from an application, or keyboard.
- Provide logging of error, performance and state change information.
- Baseline system messages using the System Message Catalog to include message and help text.

Performance Requirements from SLS

- None assigned for Thor

Other Constraint Manager Related Requirements from SLS

2.2.4.3.8 The RTPS shall provide a Constraint Monitor Viewer which provides a mechanism for asserting and viewing constraints against measurement FDs for Constraint Monitor purposes only.

2.2.5.4.1 The RTPS shall provide the capability to monitor Measurement FDs at the rate the data changes and determine when predefined constraint limits are exceeded or constraint conditions are met.

2.2.5.4.1 The RTPS shall provide the capability for multiple (TBD number) users and system or user applications to request notification of constraint events for each Measurement FD.

2.2.5.4.1 The RTPS shall allow an application to set more than one set of limits on any Measurement FD.

2.2.5.4.2 The RTPS shall provide the capability for each user, and system or user application requesting constraint notification to specify the limits/condition under which they will be notified.

2.2..5.4.2 The RTPS shall provide the capability to monitor measurement data (both converted count data or calibrated engineering units) for out of limits excursions and notify registered users when any of the following conditions occur:

1. *N samples in a row that meet the constraint*
2. *N samples in a given time that meet the constraint*
3. *There is less than a specified time between constraint events*

2.2.5.4.3 CLCS shall provide the user the capability for any user and system or user application to create new constraints from an application, keyboard, *or predefined file*, determine and/or view the current constraints and their algorithms, modify the list of constraints, and select the algorithms relating to them.

2.2.5.4.4 The RTPS Constraint Management function shall be fault tolerant.

2.2.5.4.5 *The RTPS shall provide the capability to test for or view constraint violations at a summary level (e.g. Launch Commit Criteria or OMRSD).*

2.2..5.7.4 RTPS shall utilize Constraint Management for constraint monitoring for a Test Application Script.